Polylogarithmic Approximation
for Generalized Minimum Manhattan Networks

Aparna Das[1], Krzysztof Fleszar[2], Stephen Kobourov[1], Joachim Spoerhase[2], Sankar Veeramoni[1],
and Alexander Wolff[2]

[1] Department of Computer Science, University of Arizona, Tucson, AZ, U.S.A.
[2] Lehrstuhl I, Institut für Informatik, Universität Würzburg, Germany

Abstract. Given a set of n terminals, which are points in d-dimensional Euclidean space, the *minimum Manhattan network problem* (MMN) asks for a minimum-length rectilinear network that connects each pair of terminals by a Manhattan path, that is, a path consisting of axis-parallel segments whose total length equals the pair's Manhattan distance. Even for $d = 2$, the problem is NP-hard, but constant-factor approximations are known. For $d \geq 3$, the problem is APX-hard; it is known to admit, for any $\varepsilon > 0$, an $O(n^{\varepsilon})$-approximation.

In the *generalized minimum Manhattan network problem* (GMMN), we are given a set R of n terminal *pairs*, and the goal is to find a minimum-length rectilinear network such that each pair in R is connected by a Manhattan path. GMMN is a generalization of both MMN and the well-known rectilinear Steiner arborescence problem (RSA). So far, only special cases of GMMN have been considered.

We present an $O(\log^{d+1} n)$-approximation algorithm for GMMN (and, hence, MMN) in $d \geq 2$ dimensions and an $O(\log n)$-approximation algorithm for 2D. We show that an existing $O(\log n)$-approximation algorithm for RSA in 2D generalizes easily to $d > 2$ dimensions.

1 Introduction

Given a set of terminals, which are points in $\mathbb{R}^d$, the *minimum Manhattan network problem* (MMN) asks for a minimum-length rectilinear network that connects every pair of terminals by a Manhattan path (*M-path*, for short), that is, a path consisting of axis-parallel segments whose total length equals the pair's Manhattan distance.

In the *generalized minimum Manhattan network problem* (GMMN), we are given a set R of n unordered terminal *pairs*, and the goal is to find a minimum-length rectilinear network such that every pair in R is *M-connected*, that is, connected by an M-path. GMMN is a generalization of MMN since R may contain all possible pairs of terminals. Figure 1 depicts examples of both network types.

We remark that in this paper we define n to be the number of terminal *pairs* of a GMMN instance, whereas previous works on MMN defined n to be the number of *terminals*.

Two-dimensional MMN (2D-MMN) naturally arises in VLSI circuit layout [GLN01], where a set of terminals (such as gates or transistors) needs to be interconnected by rectilinear paths (wires). Minimizing the cost of the network (which means minimizing the total wire length) is desirable in terms of energy consumption and signal interference. The additional requirement that the terminal pairs are connected by *shortest* rectilinear paths aims at decreasing the interconnection delay (see Cong et al. [CLZ93] for a discussion in the context of rectilinear Steiner arborescences, which have the same additional requirement; see definition below).

(a) an MMN for $\{a, b, c, d, e, f\}$

(b) a GMMN for $\{(a,b), (c,d), (e,f)\}$
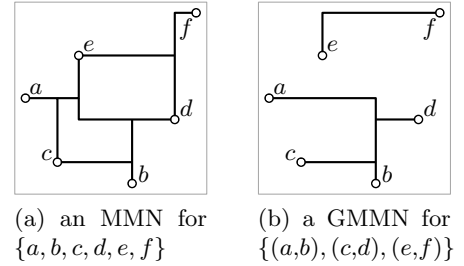
Fig. 1: MMN versus GMMN in 2D.

Manhattan networks also arise in the area of geometric spanner networks. Specifically, a minimum Manhattan network can be thought of as the cheapest spanner under the L_1-norm for given a set of points (allowing Steiner points). Spanners, in turn, have numerous applications in network design, distributed algorithms, and approximation algorithms.

MMN requires a Manhattan path between every terminal pair. This assumption is, however, not always reasonable. Specifically in VLSI design a wire connection is necessary only for a, often comparatively small, subset of terminal pairs, which may allow for substantially cheaper circuit layouts. In this scenario, GMMN appears to be a more realistic model than MMN.

Previous Work. MMN was introduced by Gudmundsson et al. [GLN01] who gave 4- and 8-approximation algorithms for 2D-MMN running in $O(n^3)$ and $O(n \log n)$ time, respectively. The currently best known approximation algorithms for 2D-MMN have ratio 2; they were obtained independently by Chepoi et al. [CNV08] using an LP-based method, by Nouioua [Nou05] using a primal-dual scheme, and by Guo et al. [GSZ11] using a greedy approach. The complexity of 2D-MMN was settled only recently by Chin et al. [CGS11]; they proved the problem NP-hard. It is not known whether 2D-MMN is APX-hard.

Less is known about MMN in dimensions greater than 2. Muñoz et al. [MSU09] proved that 3D-MMN is NP-hard to approximate within a factor of 1.00002. They also gave a constant-factor approximation algorithm for a, rather restricted, special case of 3D-MMN. More recently, Das et al. described the first approximation algorithm for MMN in arbitrary, fixed dimension with a ratio of $O(n^\varepsilon)$ for any $\varepsilon > 0$ [DGK+11].

GMMN was defined by Chepoi et al. [CNV08] who asked whether 2D-GMMN admits an $O(1)$-approximation. Apart from the formulation of this open problem, only special cases of GMMN (such as MMN) have been considered so far.

One such special case (other than MMN) that has received significant attention in the past is the *rectilinear Steiner arborescence problem* (RSA). Here, we are given n terminals lying in the first quadrant and the goal is to find a minimum-length rectilinear network that M-connects every terminal to the origin o. Hence, RSA is the special case of GMMN where o is considered a (new) terminal and the set of terminal pairs contains, for each terminal $t \neq o$, only the pair (o, t). RSA was introduced by Nastansky et al. [NSS74] and has mainly been studied in 2D. 2D-RSA is NP-hard [SS00]. Rao et al. [RSHS92] gave a 2-approximation algorithm for 2D-RSA. They also provided a conceptually simpler $O(\log n)$-approximation algorithm based on rectilinear Steiner trees. That algorithm generalizes quite easily to dimensions $d > 2$ (as we show in the appendix). Lu et al. [LR00] and, independently, Zachariasen [Zac00] described polynomial-time approximation schemes (PTAS) for RSA, both based on Arora's technique [Aro03]. Zachariasen pointed out that his PTAS can be generalized to the all-quadrant version of RSA but that it seems difficult to extend the approach to higher dimensions.

Our Contribution. Our main result is an $O(\log^{d+1} n)$-approximation algorithm for GMMN (and, hence, MMN) in d dimensions. For the sake of simplicity, we first present our approach in 2D (see Section 2) and then show how it can be generalized to higher dimensions (see Section 3). We also provide an improved and technically more involved $O(\log n)$-approximation for the special case of 2D-GMMN, but this approach does not seem to generalize to higher dimensions; see Section 4. To the best of our knowledge, we present the first approximation algorithms for GMMN. Our result for 2D is not quite the constant-factor approximation that Chepoi et al. were asking for, but it is a considerable step into that direction. Note that the poly-logarithmic ratio of our algorithm

for GMMN in $d \geq 3$ dimensions constitutes an exponential improvement upon the previously only known approximation algorithm, which solves the special case MMN, with a ratio of $O(n^\varepsilon)$ for any $\varepsilon > 0$ [DGK+11].

Our algorithm for GMMN is based on divide and conquer. We identify each terminal pair with its d-dimensional bounding box. Consequently, we consider R a set of d-dimensional boxes. We use $(d-1)$-dimensional hyperplanes to partition R recursively into sub-instances. The base case of our partition scheme consists of GMMN instances where all boxes contain a common point. We solve the resulting special case of GMMN by reducing it to RSA. We have postponed the running-time analysis to Appendix B.

2 Polylogarithmic Approximation for Two Dimensions

In this section, we present an $O(\log^2 n)$-approximation algorithm for 2D-GMMN. The algorithm consists of a *main algorithm* that recursively subdivides the input instance into instances of so-called *x-separated* GMMN; see Section 2.1. We prove that the instances of *x*-separated GMMN can be solved independently by paying a factor of $O(\log n)$ in the approximation ratio. Then we show how to approximate *x*-separated GMMN within ratio $O(\log n)$; see Section 2.2. This yields an overall ratio of $O(\log^2 n)$.

2.1 Main Algorithm

Our approximation algorithm is based on divide and conquer. Let R be the set of terminal pairs that are to be M-connected. We identify each terminal pair with its bounding box, that is, the smallest axis-aligned rectangle that contains both terminals. As a consequence of this, we consider R a set of rectangles. Let m_x be the median in the multiset of the *x*-coordinates of terminals. We identify m_x with the vertical line at $x = m_x$.

Our algorithm divides R into three subsets $R_{\text{left}}, R_{\text{mid}}$, and R_{right}. The set R_{left} consists of all rectangles that lie *completely* to the left of the vertical line m_x. Similarly, the set R_{right} consists of all rectangle that lie *completely* to the right of m_x. The set R_{mid} consists of all rectangles that intersect m_x.

We consider the sets $R_{\text{left}}, R_{\text{mid}}$, and R_{right} as separate instances of GMMN and apply our algorithm recursively to R_{left} and to R_{right}. The union of the two resulting networks is a rectilinear network that M-connects all terminal pairs in $R_{\text{left}} \cup R_{\text{right}}$.

It remains to M-connect the pairs in R_{mid}. We call an GMMN instance (such as R_{mid}) *x-separated* if there is a vertical line (in our case m_x) that intersects every rectangle. We exploit this property to design a simple $O(\log n)$-approximation algorithm for *x*-separated GMMN; see Section 2.2. Later, in Section 4, we improve upon this and describe an $O(1)$-approximation algorithm for *x*-separated GMMN.

To analyze the performance of our main algorithm, let $\rho(n)$ denote the algorithm's worst-case approximation ratio for instances with n terminal pairs. Now assume that our input instance R is a worst case. More precisely, the cost of the solution of our algorithm *equals* $\rho(n) \cdot \text{OPT}$, where OPT denotes the cost of an optimum solution N_{opt} to R. Let N_{left} and N_{right} be the parts of N_{opt} to the left and to the right of m_x, respectively.

Due to the choice of m_x, at most n terminals lie to the left of m_x. Therefore, R_{left} contains at most $n/2$ terminal pairs. Since N_{left} is a feasible solution to R_{left}, we conclude that the cost of the solution to R_{left} computed by our algorithm is bounded by $\rho(n/2) \cdot \|N_{\text{left}}\|$, where $\| \cdot \|$ measures

the length of a network. Analogously, the cost of the solution computed for R_{right} is bounded by $\rho(n/2) \cdot \|N_{\text{right}}\|$. Now we assume that we can approximate x-separated instances with a ratio of $\rho_x(n)$. Since N_{opt} is also a feasible solution to the x-separated instance R_{mid}, we can compute a solution of cost $\rho_x(n) \cdot \text{OPT}$ for R_{mid}.

Therefore, we can bound the total cost of our algorithm's solution N to R by

$$\rho(n) \cdot \text{OPT} = \|N\| \le \rho(n/2) \cdot (\|N_{\text{left}}\| + \|N_{\text{right}}\|) + \rho_x(n) \cdot \text{OPT}.$$

Note that this inequality does not necessarily hold if R is *not* a worst case since then $\rho(n) \cdot \text{OPT} > \|N\|$. The networks N_{left} and N_{right} are separated by m_x, hence they are edge disjoint and $\|N_{\text{left}}\| + \|N_{\text{right}}\| \le \text{OPT}$. This yields the recurrence $\rho(n) \le \rho(n/2) + \rho_x(n)$, which resolves to $\rho(n) = \log n \cdot \rho_x(n)$. Let's summarize this discussion.

Lemma 1. *If x-separated 2D-GMMN admits a $\rho_x(n)$-approximation, 2D-GMMN admits a $(\rho_x(n) \cdot \log n)$-approximation.*

Combining this lemma with our $O(\log n)$ approximation algorithm for x-separated instances described below, we obtain the following intermediate result.

Theorem 1. *2D-GMMN admits an $O(\log^2 n)$-approximation.*

2.2 Approximating x-Separated Instances

In this section, we describe a simple algorithm for approximating x-separated 2D-GMMN instances with a ratio of $O(\log n)$. Let R be our input. Since R is x-separated, all rectangles in R intersect a common vertical line. W.l.o.g., this is the y-axis.

The algorithm works as follows. Analogously to the main algorithm presented in Section 2.1, we recursively subdivide the x-separated input instance, but this time according to the *y-coordinate*. As a result of this, the input instance R is decomposed into *y-separated* sub-instances. Moreover, since each of these sub-instances is (as a subset of R) already x-separated, we call these instances *xy-separated*. In Section 2.3, we give a specialized algorithm for xy-separated instances.

Let $\rho_x(n)$ be the ratio of our algorithm for approximating x-separated GMMN instances and let $\rho_{xy}(n)$ be the ratio of our algorithm for approximating xy-separated GMMN instances. In Section 2.3, we show that $\rho_{xy}(n) = O(1)$. Then Lemma 1 (by exchanging x- and y-coordinates) implies that $\rho_x(n) = \log n \cdot \rho_{xy}(n) = O(\log n)$.

Lemma 2. *x-separated 2D-GMMN admits an $O(\log n)$-approximation.*

2.3 Approximating xy-Separated Instances

It remains to show that xy-separated GMMN instances can be approximated within a constant ratio. Let R be such an instance. We assume, w.l.o.g., that it is the x- and the y-axis that intersect all rectangles in R, that is, all rectangles contain the origin. Let N_{opt} be an optimum solution to R. Let N be the union of N_{opt} with the projections of N_{opt} to the x-axis and to the y-axis. The total length of N is $\|N\| \le 2 \cdot \text{OPT} = O(\text{OPT})$ since every line segment of N_{opt} is projected either to the x-axis or to the y-axis but not to both. The crucial fact about N is that this network contains, for every terminal t in R, an M-path from t to the *origin o*. In other words, N is a feasible solution to the RSA instance of M-connecting every terminal in R to o.

4

To see this, consider an arbitrary terminal pair $(t, t') \in R$. Let Π be an M-path connecting t and t' in N_{opt}; see Fig. 2. Note that, since the bounding box of (t, t') contains o, Π intersects both x- and y-axis. To obtain an M-path from t to o, we follow Π from t to t' until Π crosses one of the axes. From that point on, we follow the *projection* of Π on this axis. We reach o when Π crosses the other axis; see the dotted path in Fig. 2. Analogously, we obtain an M-path from t' to o.
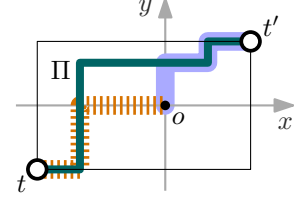
Fig. 2: Network N connects all terminals to origin o.

Let T be the set of terminals in R. We have shown above that there is a feasible solution N of cost $O(\text{OPT})$ to the RSA instance with terminal set T. There is a PTAS for RSA in two dimensions [Zac00, LR00]. Using this PTAS, we can efficiently *compute* a feasible RSA solution N' for T of cost $O(1) \cdot \|N\| = O(\text{OPT})$. Moreover, N' is also a feasible solution to the GMMN instance R. To see this, note that N' contains, for every terminal pair $(t, t') \in R$, an M-path π from t to o and an M-path π' from o to t'. Concatenating π and π' yields an M-path from t to t' as the bounding box of (t, t') contains o. Thus we obtain the following result.

Lemma 3. *xy-separated 2D-GMMN admits a constant-factor approximation.*

3 Generalization to Higher Dimensions

In this section, we describe an $O(\log^{d+1} n)$-approximation algorithm for GMMN in d dimensions, which is a generalization of the algorithm for two dimensions presented in Section 2. Let us view this algorithm from the following perspective. In Section 2.1, we reduced GMMN to solving x-separated sub-instances at the expense of a $(\log n)$-factor in the approximation ratio (see Lemma 1). Applying the same lemma to the y-coordinates in Section 2.2, we further reduced the problem to solving xy-separated sub-instances, that is, to instances that were separated with respect to both dimensions. This caused the second $(\log n)$-factor in our approximation ratio. Finally, we were able to approximate these completely separated sub-instances within constant ratio by solving a related RSA problem (see Section 2.3).

These ideas generalize to higher dimensions. An instance R of d-dimensional GMMN is called *j-separated* for some $j \leq d$ if there exist values $s_1, \ldots, s_j$ such that, for each terminal pair $(t, t') \in R$ and for each dimension $i \leq j$, we have that s_i *separates* the i-th coordinates $x_i(t)$ of t and $x_i(t')$ of t' (meaning that either $x_i(t) \leq s_i \leq x_i(t')$ or $x_i(t') \leq s_i \leq x_i(t)$). Under this terminology, an arbitrary instance of d-dimensional GMMN is always *0-separated*.

We first show that if we can approximate j-separated GMMN with ratio $\rho_j(n)$ then we can approximate $(j-1)$-separated GMMN with ratio $\rho_j(n) \cdot \log n$; see Section 3.1. Then we show that d-separated GMMN can be approximated within a factor $\rho_d(n) = O(\log n)$; see Section 3.2. Combining these two facts and applying them inductively to arbitrary (that is, 0-separated) GMMN instances yields the following central result of our paper.

Theorem 2. *GMMN in d dimensions admits an $O(\log^{d+1} n)$-approximation.*

As a byproduct of this algorithm, we obtain an $O(\log^{d+1} n)$-approximation algorithm for *MMN* where n denotes the number of *terminals*. This holds since any MMN instance with n terminals can be considered an instance of GMMN with $O(n^2)$ terminal pairs.

Corollary 1. *MMN in d dimensions admits an $O(\log^{d+1} n)$-approximation, where n denotes the number of terminals.*

3.1 Separation

In this section, we show that if we can approximate j-separated GMMN instances with ratio $\rho_j(n)$, we can approximate $(j-1)$-separated instances with ratio $\log n \cdot \rho_j(n)$. The separation algorithm and its analysis work analogously to the main algorithm for 2D where we reduced (approximating) 2D-GMMN to (approximating) x-separated 2D-GMMN; see Section 2.1

Let R be a set of $(j-1)$-separated terminal pairs. Let m_x be the median in the multiset of the j-th coordinates of terminals. We divide R into three subsets R_{left}, R_{mid}, and R_{right}. The set R_{left} consists of all terminal pairs (t, t') such that $x_j(t), x_j(t') \leq m_x$ and R_{right} contains all terminal pairs (t, t') with $x_j(t), x_j(t') \geq m_x$. The set R_{mid} contains the remaining terminal pairs, all of which are separated by the hyperplane $x_j = m_x$. We apply our algorithm recursively to R_{left} and R_{right}. The union of the resulting networks is a rectilinear network that M-connects all terminal pairs $R_{\text{left}} \cup R_{\text{right}}$.

In order to M-connect the pairs in R_{mid}, we apply an approximation algorithm for j-separated GMMN of ratio $\rho_j(n)$. Note that the instance R_{mid} is in fact j-separated by construction. The analysis of the resulting algorithm for $(j-1)$-separated GMMN is analogous to the 2D-case (see Section 2.1) and is therefore omitted.

Theorem 3. *Let $1 \leq j \leq d$. If j-separated GMMN admits a $\rho_j(n)$-approximation, then $(j-1)$-separated GMMN admits a $(\rho_j(n) \cdot \log n)$-approximation.*

3.2 Approximating d-Separated Instances

In this section, we show that we can approximate instances of d-separated GMMN within a ratio of $O(\log n)$ by reducing the problem to RSA. Let R be a d-separated instance and let T be the set of all terminals in R. As R is d-separated, all bounding boxes defined by terminal pairs in R contain a common point, which is, w.l.o.g., the origin.

As in the two-dimensional case (see Section 2.3), we M-connect all terminals to the origin by solving an RSA instance with terminal set T. This yields a feasible GMMN solution to R since for each pair $(t, t') \in R$ there is an M-path from t to the origin and an M-path from the origin to t'. The union of these paths is an M-path from t to t' since the origin is contained in the bounding box of (t, t').

Rao et al. [RSHS92] presented an $O(\log |T|)$-approximation algorithm for 2D-RSA, which generalizes, in a straight-forward manner, to d-dimensional RSA; see Appendix A for details. Hence, we can use the algorithm of Rao et al. to efficiently compute a feasible GMMN solution. The following lemma shows that this solution is in fact an $O(\log n)$-approximation. The proof is similar to the proof of Lemma 7 in the paper of Das et al. [DGK$^+$11].

Lemma 4. *d-separated GMMN admits an $O(\log n)$-approximation for any fixed number d of dimensions.*

Proof. Below we show that there is a solution of cost $O(\text{OPT})$ to the RSA instance connecting T to the origin. Observing that $|T| \leq 2n$ and using our extension of the $O(\log |T|)$-approximation algorithm of Rao et al. (see Appendix A), we can efficiently compute a feasible GMMN solution of cost $O(\text{OPT} \cdot \log n)$.

We now show that there is an RSA solution of cost $O(\text{OPT})$. Let N_{opt} be an optimal GMMN solution to R and let N be the projection of N_{opt} onto all subspaces that are spanned by some

subset of the coordinate axes. Since there are 2^d such subspaces, which is a constant for fixed d, the cost of N is $O(\text{OPT})$.

It remains to show that N M-connects all terminals to the origin, that is, N is a feasible solution to the RSA instance. First, note that $N_{\text{opt}} \subseteq N$ since we project on the d-dimensional space, too. Now consider an arbitrary terminal pair (t, t') in R and an M-path π in N_{opt} that M-connects t and t'. Starting at t, we follow π until we reach the first point p_1 where one of the coordinates becomes zero. W.l.o.g., $x_1(p_1) = 0$. Clearly π contains such a point as the bounding box of (t, t') contains the origin. We observe that p_1 lies in the subspace spanned by the $d-1$ coordinate axes $x_2, \ldots, x_d$. From p_1 on we follow the projection of π onto this subspace until we reach the first point p_2 where another coordinate becomes zero; w.l.o.g., $x_2(p_2) = 0$. Hence, p_2 has at least two coordinates that are zero, that is, p_2 lies in a subspace spanned by only $d-2$ coordinate axes. Iteratively, we continue following the projections of π onto subspaces with a decreasing number of dimensions until every coordinate is zero, that is, we have reached the origin. An analogous argument shows that N also contains an M-path from t' to the origin. □

4 Improved Algorithm for Two Dimensions

In this section, we show that 2D-GMMN admits an $O(\log n)$-approximation, which improves upon the $O(\log^2 n)$-result of Section 2. To this end, we develop a $(6 + \varepsilon)$-approximation algorithm for x-separated 2D-GMMN, for any $\varepsilon > 0$. While the algorithm is simple, its analysis turns out to be quite intricate. In Appendix C, we show tightness. Using Lemma 1, our new subroutine for the x-separated case yields the following.

Theorem 4. *2D-GMMN admits a* $((6 + \varepsilon) \cdot \log n)$*-approximation.*

Let R be the set of terminal pairs of an x-separated instance of 2D-GMMN. We assume, w.l.o.g., that each terminal pair $(l, r) \in R$ is separated by the y-axis, that is, $x(l) < 0 \leq x(r)$. Let N_{opt} be an optimum solution to R. Let OPT_{ver} and OPT_{hor} be the total costs of the vertical and horizontal segments in N_{opt}, respectively. Hence, $\text{OPT} = \text{OPT}_{\text{ver}} + \text{OPT}_{\text{hor}}$. We first compute a set S of horizontal line segments of total cost $O(\text{OPT}_{\text{hor}})$ such that each rectangle in R is *stabbed* by some line segment in S; see Sections 4.1 and 4.2. Then we M-connect the terminals to the y-axis so that the resulting network, along with the affected part of the y-axis and the stabbing S, forms a feasible solution to R of cost $O(\text{OPT})$; see Section 4.3.

4.1 Stabbing the Right Part

We say that a horizontal line segment h *stabs* an axis-aligned rectangle r if h intersects the boundary of r twice. A set of horizontal line segments is a *stabbing* of a set of axis-aligned rectangles if each rectangle is stabbed by some line segment. For any geometric object, let its *right part* be its intersection with the closed half plane to the right of the y-axis. For a *set* of objects, let its right part be the set of the right parts of the objects. Let R^+ be the right part of R, let N^+ be the right part of N_{opt}, and let N_{hor}^+ be the set of horizontal line segments in N^+. In this section, we show how to construct a stabbing of R^+ of cost at most $2 \cdot \|N_{\text{hor}}^+\|$.

For $x' \geq 0$, let $\ell_{x'}$ be the vertical line at $x = x'$. Our algorithm performs a left-to-right sweep starting with ℓ_0. Note that, for every $x \geq 0$, the intersection of R^+ with ℓ_x forms a set $\mathcal{I}_x$ of intervals. The intersection of N_{hor}^+ ($\bigcup N_{\text{hor}}^+$ to be precise) with ℓ_x is, at any time, a set of points

that constitutes a *piercing* for $\mathcal{I}_x$, that is, every interval in $\mathcal{I}_x$ contains a point in $\ell_x \cap N_{\text{hor}}^+$. Note that $\|N_{\text{hor}}^+\| = \int |\ell_x \cap N_{\text{hor}}^+| dx$.

We imagine that we continuously move ℓ_x from $x = 0$ to the right. At any time, we maintain an inclusion-wise minimal piercing P_x of $\mathcal{I}_x$. With increasing x, we only *remove* points from P_x; we never add points. This ensures that the traces of the points in P_x form horizontal line segments that all touch the y-axis. These line segments form our stabbing of R^+.

The algorithm proceeds as follows. It starts at $x := 0$ with an arbitrary minimal piercing P_0 of $\mathcal{I}_0$. Note that we can even compute an optimum piercing P_0. We must adapt P_x whenever $\mathcal{I}_x$ changes. With increasing x, $\mathcal{I}_x$ decreases inclusion-wise since all rectangles in R^+ touch the y-axis. So it suffices to adapt the piercing P_x at *event points*; x is an event point if and only if x is the x-coordinate of a right edge of a rectangle in R^+.

Let x' and x'' be consecutive event points. Let x be such that $x' < x \leq x''$. Note that $P_{x'}$ is a piercing for $\mathcal{I}_x$ since $\mathcal{I}_x \subset \mathcal{I}_{x'}$. The piercing $P_{x'}$ is, however, not necessarily *minimal* w.r.t. $\mathcal{I}_x$. When the sweep line passes x', we therefore have to drop some of the points in $P_{x'}$ in order to obtain a new minimal piercing. This can be done by iteratively removing points from $P_{x'}$ such that the resulting set still pierces $\mathcal{I}_x$. We stop at the last event point (afterwards, $\mathcal{I}_x = \emptyset$) and output the traces of the piercing.

It is clear that the algorithm produces a stabbing of R^+; see the thick solid line segments in Fig. 3a. The following lemma is crucial to prove the overall cost of the stabbing.

Lemma 5. *For any $x \geq 0$, it holds that $|P_x| \leq 2 \cdot |\ell_x \cap N_{\text{hor}}^+|$.*

Proof. Since P_x is a minimal piercing, there exists, for every $p \in P_x$, a *witness interval* $I_p \in \mathcal{I}_x$ that is pierced by p but not by $P_x \setminus \{p\}$. Otherwise we could remove p from P_x, contradicting the minimality of P_x.

Now we show that an arbitrary point q on ℓ_x is contained in the witness intervals of at most two points in P_x. Assume, for the sake of contradiction, that q is contained in the witness intervals of points $p, p', p'' \in P_x$ with strictly increasing y-coordinates. Suppose that q lies above p'. But then the witness interval I_p of p, which contains p and q, must also contain p', contradicting the definition of I_p. The case q below p' is symmetric.

Recall that $\ell_x \cap N_{\text{hor}}^+$ is a piercing of $\mathcal{I}_x$ and, hence, of the $|P_x|$ many witness intervals. Since every point in $\ell_x \cap N_{\text{hor}}^+$ pierces at most two witness intervals, the lemma follows. $\square$
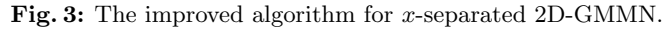
Observe that the cost of the stabbing is $\int |P_x| dx$. By the above lemma, the cost of the stabbing can be bounded by $\int |P_x| dx \leq \int 2 \cdot |\ell_x \cap N_{\text{hor}}^+| dx = 2 \cdot \|N_{\text{hor}}^+\|$, which proves the following lemma.

Lemma 6. *Given a set R of rectangles intersecting the y-axis, we can compute a set of horizontal line segments of cost at most $2 \cdot \text{OPT}_{\text{hor}}$ that stabs R^+.*

4.2 Stabbing Both Parts

We now detail how we construct a stabbing of R. To this end we apply Lemma 6 to compute a stabbing S^- of cost at most $2 \cdot \|N_{\text{hor}}^-\|$ for the left part R^- of R and a stabbing S^+ of cost at most $2 \cdot \|N_{\text{hor}}^+\|$ for the right part R^+. Note that $S^- \cup S^+$ is not necessarily a stabbing for R since there can be rectangles that are not *completely* stabbed by one segment. To overcome this difficulty, we mirror S^- and S^+ to the respective other side of the y-axis; see Fig. 3a. The total cost of the

(a) S^+ stabs R^+, and S stabs R.

(b) $N = A_{\mathrm{up}} \cup A_{\mathrm{down}} \cup S$ is feasible for R.

(c) $N_{\mathrm{opt}} \cup \{I\}$ is feasible for RSA instances $(L, \mathrm{top}(I))$, $(H, \mathrm{bot}(I))$.

Fig. 3: The improved algorithm for x-separated 2D-GMMN.

resulting set S of horizontal line segments is at most $4(\|N^-_{\mathrm{hor}}\| + \|N^+_{\mathrm{hor}}\|) = 4 \cdot \mathrm{OPT}_{\mathrm{hor}}$. The set S stabs R since, for every rectangle $r \in R$, the larger among its two (left and right) parts is stabbed by some segment s and the smaller part is stabbed by the mirror image s' of s. Hence, r is stabbed by the line segment $s \cup s'$. Let us summarize.

Lemma 7. *Given a set R of rectangles intersecting the y-axis, we can compute a set of horizontal line segments of cost at most $4 \cdot \mathrm{OPT}_{\mathrm{hor}}$ that stabs R.*

4.3 Connecting Terminals and Stabbing

We assume that the union of the rectangles in R is connected. Otherwise we apply our algorithm separately to each subset of R that induces a connected component of $\bigcup R$. Let I be the line segment that is the intersection of the y-axis with $\bigcup R$. Let $\mathrm{top}(I)$ and $\mathrm{bot}(I)$ be the top and bottom endpoints of I, respectively. Let L be the set containing every terminal t with $(t, t') \in R$ and $y(t) \le y(t')$. Symmetrically, let H be the set containing every terminal t with $(t, t') \in R$ and $y(t) > y(t')$. Note that L and H are not necessarily disjoint.

Using a PTAS for RSA [LR00, Zac00], we compute a near-optimal RSA network A_{up} connecting the terminals in L to $\mathrm{top}(I)$ and a near-optimal RSA network A_{down} connecting the terminals in H to $\mathrm{bot}(I)$. Then we return the network $N = A_{\mathrm{up}} \cup A_{\mathrm{down}} \cup S$, where S is the stabbing computed by the algorithm in Section 4.2.

We now show that this network is a feasible solution to R. Let $(l, h) \in R$. W.l.o.g., $l \in L$ and $h \in H$. Hence, A_{up} contains a path π_l from l to $\mathrm{top}(I)$, see Fig. 3b. This path starts inside the rectangle (l, h) and leaves it through its top edge. Before leaving (l, h), the path intersects a line segment s in S that stabs (l, h). This line segment is also intersected by the path π_h in A_{down} that connects h to $\mathrm{bot}(I)$. Hence, walking along π_l, s, and π_h brings us in a monotone fashion from l to h.

Now, let us analyze the cost of N. Clearly, the projection of N_{opt} onto the y-axis yields the line segment I. Hence, $|I| \le \mathrm{OPT}_{\mathrm{ver}}$. Observe that $N_{\mathrm{opt}} \cup \{I\}$ constitutes a solution to the RSA instance $(L, \mathrm{top}(I))$ connecting all terminals in L to $\mathrm{top}(I)$ and to the RSA instance $(H, \mathrm{bot}(I))$ connecting all terminals in H to $\mathrm{bot}(I)$. This holds since, for each terminal pair, its M-path π in N_{opt} crosses the y-axis in I; see Fig. 3c. Since A_{up} and A_{down} are near-optimal solutions to these RSA instances, we obtain, for any $\varepsilon > 0$, that $\|A_{\mathrm{up}}\| \le (1 + \varepsilon) \cdot \|N_{\mathrm{opt}} \cup I\| \le (1 + \varepsilon) \cdot (\mathrm{OPT} + \mathrm{OPT}_{\mathrm{ver}})$ and analogously $\|A_{\mathrm{down}}\| \le (1 + \varepsilon) \cdot (\mathrm{OPT} + \mathrm{OPT}_{\mathrm{ver}})$.

By Lemma 7, we have that $\|S\| \leq 4 \cdot \mathrm{OPT}_{\mathrm{hor}}$. Assuming $\varepsilon \leq 1$, this yields

$$
\begin{aligned}
\|N\| &= \|A_{\mathrm{up}}\| + \|A_{\mathrm{down}}\| + \|S\| \\
&\leq (2 + 2\varepsilon) \cdot (\mathrm{OPT} + \mathrm{OPT}_{\mathrm{ver}}) + 4 \cdot \mathrm{OPT}_{\mathrm{hor}} \\
&\leq (2 + 2\varepsilon) \cdot \mathrm{OPT} + 4 \cdot (\mathrm{OPT}_{\mathrm{ver}} + \mathrm{OPT}_{\mathrm{hor}}) \\
&= (6 + \varepsilon') \cdot \mathrm{OPT}
\end{aligned}
$$

for $\varepsilon' = \varepsilon/2$, which we can make arbitrarily small by making ε arbitrarily small. We summarize our result as follows.

Lemma 8. *x-separated 2D-GMMN admits, for any $\varepsilon > 0$, a $(6 + \varepsilon)$-approximation.*

5 Conclusion and Open Problems

We have presented an $O(\log^{d+1} n)$-approximation algorithm for d-dimensional GMMN, which implies the same ratio for MMN. Prior to our work, no approximation algorithm for GMMN was known. For $d \geq 3$, our result is a significant improvement over the ratio of $O(n^\varepsilon)$ of the only approximation algorithm for d-dimensional MMN known so far.

In 2D, there is still quite a large gap between the currently best approximation ratios for MMN and GMMN. Whereas we have presented an $O(\log n)$-approximation algorithm for 2D-GMMN, 2D-MMN admits 2-approximations [CNV08, GSZ11, Nou05]—but is 2D-GMMN really harder to approximate than 2D-MMN? Indeed, given that GMMN is more general than MMN, it may be possible to derive stronger non-approximability results for GMMN. So far, the only such result is that 3D-MMN cannot be approximated beyond a factor of 1.00002 [MSU09].

Concerning the positive side, for $d \geq 3$, a constant-factor approximation for d-dimensional RSA would shave off a factor of $O(\log n)$ from the current ratio for d-dimensional GMMN. This may be in reach given that 2D-RSA admits even a PTAS [LR00, Zac00]. Alternatively, a constant-factor approximation for $(d - k)$-separated GMMN for some $k \leq d$ would shave off a factor of $O(\log^k n)$ from the current ratio for d-dimensional GMMN.

Acknowledgments

References

[Aro98] Sanjeev Arora. Polynomial time approximation schemes for euclidean traveling salesman and other geometric problems. *J. ACM*, 45(5):753–782, 1998.

[Aro03] Sanjeev Arora. Approximation schemes for NP-hard geometric optimization problems: A survey. *Math. Program.*, 97(1–2):43–69, 2003.

[CGS11] Francis Chin, Zuyu Guo, and He Sun. Minimum Manhattan network is NP-complete. *Discrete Comput. Geom.*, 45:701–722, 2011.

[CLZ93] Jason Cong, Kwok-Shing Leung, and Dian Zhou. Performance-driven interconnect design based on distributed RC delay model. In *Proc. 30th IEEE Conf. Design Automation (DAC'93)*, pages 606–611, 1993.

[CNV08] Victor Chepoi, Karim Nouioua, and Yann Vaxès. A rounding algorithm for approximating minimum Manhattan networks. *Theor. Comput. Sci.*, 390(1):56–69, 2008.

[DGK⁺11] Aparna Das, Emden R. Gansner, Michael Kaufmann, Stephen Kobourov, Joachim Spoerhase, and Alexander Wolff. Approximating minimum Manhattan networks in higher dimensions. In Camil Demetrescu and Magnús M. Halldórsson, editors, *Proc. 19th Annu. Europ. Symp. on Algorithms (ESA'11)*, volume 6942 of *LNCS*, pages 49–60. Springer, 2011.

[GLN01] Joachim Gudmundsson, Christos Levcopoulos, and Giri Narasimhan. Approximating a minimum Manhattan network. *Nordic J. Comput.*, 8:219–232, 2001.

[GSZ11] Zeyu Guo, He Sun, and Hong Zhu. Greedy construction of 2-approximate minimum Manhattan networks. *Internat. J. Comput. Geom. Appl.*, 21(3):331–350, 2011.

[LR00] Bing Lu and Lu Ruan. Polynomial time approximation scheme for the rectilinear Steiner arborescence problem. *J. Comb. Optim.*, 4(3):357–363, 2000.

[MSU09] Xavier Muñoz, Sebastian Seibert, and Walter Unger. The minimal Manhattan network problem in three dimensions. In Sandip Das and Ryuhei Uehara, editors, *Proc. 3rd Int. Workshop Algorithms Comput. (WALCOM'09)*, volume 5431 of *LNCS*, pages 369–380. Springer, 2009.

[Nou05] Karim Nouioua. *Enveloppes de Pareto et Réseaux de Manhattan: Caractérisations et Algorithmes*. PhD thesis, Université de la Méditerranée, 2005.

[NSS74] Ludwig Nastansky, Stanley M. Selkow, and Neil F. Stewart. Cost-minimal trees in directed acyclic graphs. *Zeitschrift Oper. Res.*, 18(1):59–67, 1974.

[RSHS92] Sailesh Rao, P. Sadayappan, Frank Hwang, and Peter Shor. The rectilinear Steiner arborescence problem. *Algorithmica*, 7:277–288, 1992.

[SS00] Weiping Shi and Chen Su. The rectilinear Steiner arborescence problem is NP-complete. In *Proc 11th Annu. ACM-SIAM Symp. Discrete Algorithms (SODA'00)*, pages 780–787, 2000.

[Zac00] Martin Zachariasen. On the approximation of the rectilinear Steiner arborescence problem in the plane. Unpublished manuscript, see `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.43.4529`, 2000.

Appendix

A   Solving RSA in Higher Dimensions

In this section, we show that we can approximate d-dimensional RSA with a ratio of $O(\log n)$ even in the all-orthant case where every orthant may contain terminals. In this section, n denotes the number of *terminals*. We generalize the algorithm of Rao et al. [RSHS92] who give an $O(\log n)$-approximation algorithm for the one-quadrant version of 2D-RSA.

It is not hard to verify that the $O(\log n)$-approximation algorithm of Rao et al. carries over to higher dimensions in a straightforward manner if all terminals lie in the *same* orthant. We can therefore obtain a feasible solution to the all-orthant version by applying the approximation algorithm to each orthant separately. This worsens the approximation ratio by a factor no larger than 2^d since there are 2^d orthants. Hence, we can quite easily give an $O(\log n)$-approximation algorithm for the all-orthant version since 2^d is a constant for fixed dimension d.

In what follows, we present a tailor-made approximation algorithm for the all-orthant version of d-dimensional RSA that avoids the additional factor of 2^d. Our algorithm is an adaption of the algorithm of Rao et al., and our presentation closely follows their lines, too.

Consider an instance of RSA given by a set T of terminals in $\mathbb{R}^d$ (without restriction of the orthant). Let o denote the origin. The algorithm relies on the following lemma, which we prove below.

Lemma 9. *Given a rectilinear Steiner tree B for terminal set $T \cup \{o\}$, we can find an RSA network A for T of length at most $\lceil \log_2 n \rceil \cdot \|B\|$.*

Every RSA network is also a rectilinear Steiner tree. Since the rectilinear Steiner tree problem (RST) admits a PTAS for any fixed dimension d [Aro98], we can generate a $(1 + \varepsilon)$-approximate RST network B that connects T and the origin. By means of Lemma 9, we get a $(1 + \varepsilon)\lceil \log_2 n \rceil$-approximation for the RSA instance T.

Theorem 5. *The all-orthant version of d-dimensional RSA admits a $(1+\varepsilon)\cdot\lceil \log_2 n \rceil$ approximation for any $\varepsilon > 0$.*

Our proof of Lemma 9 relies on the following technical lemma, which constitutes the main modification that we make to the algorithm of Rao et al. See Fig. 4 for an illustration.

Lemma 10. *Let t, t' be two terminals. Then we can compute in constant time a point $\min(t, t')$ and an M-path $\pi(t, t')$ from t to t' containing $\min(t, t')$ with the following property. The union of $\pi(t, t')$ with an M-path from $\min(t, t')$ to o M-connects t and t' to o.*

Proof. We start with the following simple observation. If s and s' are points and p is a point in the bounding box $B(s, s')$ of s and s', then M-connecting s to p and p to s' also M-connects s to s'.

Observe that the three bounding boxes $B(o, t)$, $B(o, t')$, and $B(t, t')$ have pairwise non-empty intersections. By the Helly property of axis-parallel d-dimensional boxes, there exists a point $\min(t, t')$ that simultaneously lies in all three boxes.

M-connecting $\min(t, t')$ with t and t' yields $\pi(t, t')$, and M-connecting $\min(t, t')$ with o yields an M-path between any two of the points t, t', o by a repeated application of the above observation. This completes the proof. $\qed$

Proof (of Lemma 9). We double the edges of B and construct a Eulerian cycle C that traverses the terminals in $T \cup \{o\}$ in some order $t_0, t_1, \ldots, t_n$. The length of C is at most $2\|B\|$ by construction. Now consider the shortcut cycle $\tilde{C}$ in which we connect consecutive terminals t_i, t_{i+1} by the M-path $\pi(t_i, t_{i+1})$ as defined in Lemma 10; we set $t_{n+1} := t_0$. Clearly $\|\tilde{C}\| \leq \|C\|$. We partition $\tilde{C}$ it into two halves; $C_0 = \{\pi(t_{2i}, t_{2i+1}) \mid 0 \leq i \leq n/2\}$ and $C_1 = \{\pi(t_{2i+1}, t_{2i+2}) \mid 0 \leq i \leq n/2 - 1\}$. For at least one of the two halves, say C_0, we have $\|C_0\| \leq \|B\|$.

We use C_0 as a partial solution and recursively M-connect the points in the set $T' := \{\min(t_{2i}, t_{2i+1}) \mid 0 \leq i \leq n/2\}$, which lie in C_0 (see Lemma 10), to the origin by an arboresence A'. Lemma 10 implies that the resulting network $A = C_0 \cup A'$ is in fact a feasible RSA solution. The length of A is at most $\|C_0\| + \|A'\| \leq \|B\| + \|A'\|$. Note that $|T'| \leq (|T| + 1)/2$.

To summarize, we have described a procedure that, given the rectilinear cycle C traversing terminal set $T \cup \{o\}$, computes a shortcut cycle $\tilde{C}$, its shorter half C_0, and a new point set T' that still has to be M-connected to the origin. We refer to this procedure as *shortcutting*.

To compute the arboresence A', observe that $\tilde{C}$ is a rectilinear cycle that traverses the points in T'. Shortcutting yield a new cycle $\tilde{C}'$ of length at most $\|\tilde{C}\| \leq \|C\|$, a half C_0' no longer than $\|B\|$, which we add to the RSA network, and a new point set T'' of cardinality $|T''| \leq |T'|/2 \leq (|T|+1)/4$, which we recursively M-connect to the origin.

We repeat the shortcutting and recurse. Each iteration halves the number of new points, so the process terminates in $O(\log n)$ iterations with a single point t. Since $\min(o, p) = o$ for any point p (see proof of Lemma 10) and our original terminal set $T \cup \{o\}$ contained o, we must have that $t = o$. This shows that the computed solution is feasible. As each iteration adds length at most $\|B\|$, we have $\|A\| \leq \lceil \log_2 n \rceil \cdot \|B\|$. $\qquad\square$

B   Running Time Analysis

We first analyze the running times of the algorithm for $d > 2$ in Section 3.

Given an instance R of 0-separated d-dimensional GMMN, the algorithm uses d recursive procedures to subdivide the problem into d-separated instances. For $j \in \{0, \ldots, d-1\}$, let $T_j(n)$ denote the running time of the j-th recursive procedure. The j-th recursive procedure takes a j-separated instance R as input and partitions it into two j-separated instances, each of size at most $|R|/2$, and one $(j + 1)$-separated instance of size at most $|R|$. The partitioning requires $O(n)$ steps for finding the median of the j-th coordinate value of terminals in R. The two j-separated instances are solved recursively and the $(j + 1)$-separated instance is solved with the $(j + 1)$-th recursive procedure. Let $T_d(n)$ denote the running time to solve a d-separated instance. As pointed out in Appendix A, we can approximate RSA in $d > 2$ dimensions by applying (an extension of) the algorithm of Rao et al. [RSHS92] to each orthant separately. This requires $O(n \log n)$ time as does the original algorithm. Thus we have

$$T_d(n) = O(n \log n)$$
$$T_j(n) = 2T_j(n/2) + T_{j+1}(n) \qquad\qquad \text{for } j \in \{0, \ldots, d-1\}$$

The running time of our overall algorithm is given by $T_0(n)$. Solving the recurrences above yields $T_0(n) = O(n \log^{d+1} n)$. The improved approximation algorithm of Theorem 5 uses Aroras PTAS [Aro98] for rectilinear Steiner trees, which worsens the running time substantially but still leads to a polynomial running time.

Now we analyze the running time of the improved algorithm of Section 4. Stabbing x-separated instances can be done with a sweep-line algorithm in $O(n \log n)$ time. The PTAS for RSA requires time $O(n^{1/\varepsilon} \log n)$ for any ε with $0 < \varepsilon \le 1$. Hence, we have that $T_1(n) = O(n^{1/\varepsilon} \log n)$, and $T_0(n) = 2T_0(n/2) + T_1(n)$. Solving the recursion yields a running time of $T_0(n) = O(n^{1/\varepsilon} \log^2 n)$ for the improved algorithm.

## C Example Showing the Tightness of Our Analysis

Observation 1 *There are infinitely many instances where the $O(\log n)$-approximation algorithm for 2D-GMMN described in Section 4 has approximation performance $\Omega(\log n)$.*

Proof. We recursively define an arrangement $A(n)$ of n rectangles each of which represents a terminal pair; the lower left and upper right corner of the rectangle. By $\alpha \cdot A(n)$ we denote the arrangement $A(n)$ but uniformly scaled in both dimensions so that it fits into an $\alpha \times \alpha$ square. Let $\varepsilon > 0$ be a sufficiently small number.

The arrangement $A(0)$ is empty. The arrangement $A(n)$ consists of a unit square S_n whose upper right vertex is the origin. We add the arrangement $A_{\text{right}} := \varepsilon \cdot A((n-1)/2)$ and place it in the first quadrant at distance ε to the origin. Finally, we add the arrangement $A_{\text{left}} := (1-\varepsilon) \cdot A((n-1)/2)$ inside the square S_n so that it does not touch the boundary of S_n. See Fig. 5 for an illustration.
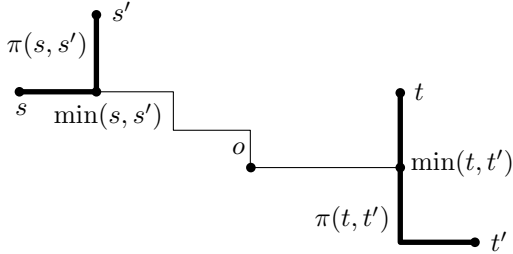
Fig. 4: Illustration of Lemma 10. For each terminal pair t, t', we compute a suitable point $\min(t, t')$ and an M-path $\pi(t, t')$ containing $\min(t, t')$. Adding an arbitrary M-path from $\min(t, t')$ to o M-connects t and t' to o.
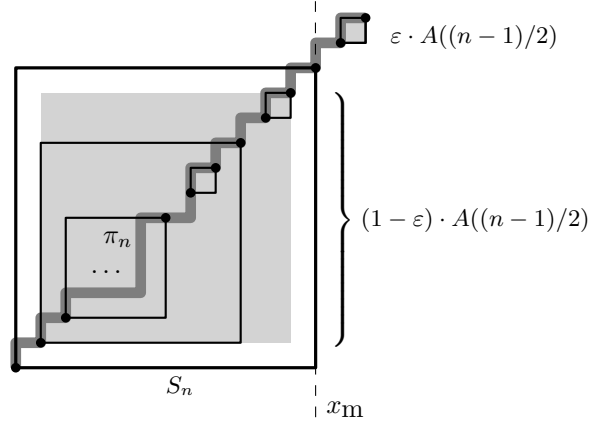
Fig. 5: Recursive construction of the arrangement $A(n)$. The gray M-path π_n shows an optimum solution. The dashed vertical line marks where the algorithm separates.

Let $\rho(n)$ denote the cost produced by our algorithm when applied to $A(n)$. Observe that our algorithm partitions $A(n)$ into subinstances $R_{\text{left}} = A_{\text{left}}$, $R_{\text{mid}} = \{S_n\}$, and $R_{\text{right}} = A_{\text{right}}$. Solving the x-separated instance R_{mid} by our stabbing subroutine costs 1. Let $\rho(n)$ be the cost of the solution to $A(n)$ that our algorithm computes. Recursively solving R_{left} costs $(1-\varepsilon) \cdot \rho((n-1)/2)$. Recursively solving R_{right} costs $\varepsilon \cdot \rho((n-1)/2)$. Hence, the cost of the solution of our algorithm is $\rho(n) \ge 1 + \rho((n-1)/2)$. This resolves to $\rho(n) = \Omega(\log n)$.

Finally, observe that the optimum solution is a single M-path π_n of length $1 + 2\varepsilon$ going from the third to the first quadrant through the origin, see Fig. 5. $\qed$